SYSTEM, METHOD, AND COMPUTER PROGRAM PRODUCT FOR KNOWLEDGE MANAGEMENT

This is a continuation-in-part application of pending U.S. non-provisional application Ser. No. 08/921,369 filed August 29, 1997 titled "Using Hyperbolic Trees to Visualize Data Generated by Patent-Centric and Group-Oriented Data Processing," which is a continuation-in-part application of U.S. Patent No. 5,991,751 titled "System, Method, and Computer Program Product for Patent-Centric and Group-Oriented Data Processing," both of which are herein incorporated by reference in their entireties.

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] The invention is generally directed to the management of information (knowledge).

Related Art

[0002] Many systems exist for organizing documents. Such systems include file management applications (such as Windows Explorer) and document management systems.

[0003] Systems also exist for analyzing documents to some extent. Such systems include, for example, various Internet search engines for identifying documents that satisfy some supplied search criteria.

[0004] There are many more examples of existing systems for organizing documents, and systems for analyzing documents.

[0005] However, systems that effectively and efficiently manage knowledge are rare. Thus, there is a need for such systems. More particularly, there is a need for systems that enable users to organize, process, and otherwise manipulate information contained in any form (text, graphics, multimedia, applications, images, sound, etc.), and that provide diverse and flexible functionality so that users may construct work flows and processes according to their particular needs.

SUMMARY OF THE INVENTION

[0006] Briefly stated, the invention is directed to a system, method, and computer program product for managing knowledge. The knowledge that is being managed comprises documents of interest to users. Such documents may be in any form, such as but not limited to text, images, graphics, audio, video, multimedia, computer programs/applications, etc., and combinations thereof.

[0007] More particularly, the invention is directed to a computer implemented method of enabling a user to organize and analyze information in electronic form. The method operates by searching a first set of documents to thereby generate a second set of documents. The invention automatically creates a first group comprising the second set of documents. The invention analyzes the first group according to one or more analytical functions to thereby generate a third set of documents. The invention automatically creates a second group comprising the third set of documents. The invention enables selective iteration of any of these operational steps.

[0008] The invention also generates objects corresponding to process components of this work flow. The user can re-execute the work flow (process) by traversing the objects, or create a new process by modifying the objects.

[0009] Further features and advantages of the invention, as well as the structure and operation of various embodiments of the invention, are described in detail below with reference to the accompanying drawings. In the drawings, like reference numbers generally indicate identical, functionally similar, and/or structurally similar elements. Generally, the drawing in which an element first appears is indicated by the leftmost digit(s) in the corresponding reference number.

BRIEF DESCRIPTION OF THE FIGURES

[0010] The present invention will be described with reference to the accompanying drawings, wherein:

[0011] FIG. 1 is a block diagram of a computing environment that includes a knowledge management system (KMS) according to an embodiment of the invention;

[0012] FIG. 2 is a block diagram of the KMS according to an embodiment of the invention;

[0013] FIG. 3 is a block diagram of databases according to an embodiment of the invention;

[0014] FIG. 4 is a block diagram of analysis modules according to an embodiment of the invention;

[0015] FIG. 5 is a block diagram useful for implementing elements of embodiments of the invention;

[0016] FIG. 6 is an example work flow diagram according to an embodiment of the invention;

[0017] FIG. 7 is a block diagram of various searching modules according to an embodiment of the invention;

[0018] FIG. 8 is a block diagram of various citations modules according to an embodiment of the invention;

[0019] FIG. 9 is a flowchart representing the operation of an author citation module according to an embodiment of the invention;

[0020] FIGS. 10A and 10B depict another example work flow diagram according to an embodiment of the invention;

[0021] FIG. 11 indicates functionality involving objects according to an embodiment of the invention;

[0022] FIGS. 12-27 are example user interface screen shots related to objects according to an embodiment of the invention;

[0023] FIG. 28 is an example work flow diagram used to illustrate the difference between upstream objects and downstream objects;

[0024] FIG. 29 is an example flowchart used to illustrate how objects are created according to an embodiment of the invention;

[0025] FIG. 30 is an example state diagram indicating the manner in which users may create sequences of actions to thereby generate different work flows/processes;

[0026] FIG. 31 is an example work flow diagram involving the use of a clustering module according to an embodiment of the invention;

[0027] FIG. 32 is an example display produced by a cluster module according to an embodiment of the invention;

[0028] FIG. 33 is an example family tree generated by the invention;

[0029] FIG. 34 is an example work flow diagram involving the use of a relevancy visualization module according to an embodiment of the invention; and

[0030] FIG. 35 is an example display produced by a relevancy visualization module according to an embodiment of the invention; and

[0031] FIGS. 36 and 37 illustrate additional work flow examples of the invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

1. Overview of the Invention

[0032] The invention is directed to a system, method, and computer program product for managing knowledge. The knowledge that is being managed comprises documents of interest to users. Such documents may be in any form, such as but not limited to text, images, graphics, audio, video, multimedia, applications, etc., and combinations thereof.

[0033] The invention provides a number of functions for processing and manipulating knowledge. Typically, the functions that are performed are selected by users. More particularly, users select the functions that are to be performed, and the sequence in which those functions are performed. This is referred to as "work flow" or "process."

[0034] According to embodiment of the invention, work flows can be saved. In other words, the invention supports persistent work flows or processes. The invention achieves persistent processes through the use of objects. These objects can be manipulated and re-used to achieve a number of advantages.

[0035] The invention shall now be described in greater detail.

2. Description of the Knowledge Management System (KMS)

[0036] FIG. 1 is a block diagram of a system 102 according to an embodiment of the invention.

[0037] The system 102 includes a plurality of databases 116 that store documents, such as patent related information and other information (unrelated to patents).

[0038] A knowledge management system (KMS) 114 accesses and processes the information in the databases 116. In particular, the KMS 114 includes modules that are capable of semi-automatically and automatically accessing and processing the information in the databases 116 in an document-centric and/or group-oriented manner. Such processing includes, but is not limited to, reporting, analyzing, and planning.

[0039] In an embodiment, the KMS 114 is implemented at least in part using an Intellectual Property Asset Manager (IPAM), which is described in U.S. Patent No. 5,991,751, incorporated herein by reference in its entirety.

[0040] In an embodiment, the system 102 includes (but is not limited to) two types of clients, network clients 106 and web clients 104. These clients 104, 106 interact with the KMS 114 to access and process the information in the databases 116.

[0041] For example, the clients 104, 106 may request that the KMS 114 retrieve certain information, or automatically analyze certain information. The KMS 114 performs the requested tasks, and sends the results to the requesting clients 104, 106. The clients 104, 106 present these results to their respective operators, for example, and enable the operators to process the results.

[0042] Clients 104, 106 may also perform additional processing of data, such as creating a visualization of the data obtained from the KMS 114.

2.1. Example Computer Implementation

[0043] In an embodiment of the present invention, the components of the present invention shown in FIG. 1 are implemented using well known computers, such as a computer 502 shown in FIG. 5. The computer 502 can be any commercially available and well known computer capable of performing the functions described herein, such as computers available from International Business Machines, Apple, Silicon Graphics Inc., Sun, HP, Dell, Compaq, Gateway, etc.

[0044] The computer 502 includes one or more processors (also called central processing units, or CPUs), such as a processor 506. The processor 506 is connected to a communication bus 504. The computer 502 also includes a main or primary memory 508, preferably random access memory (RAM). The primary memory 508 has stored therein control logic 510 (computer software), and data 512.

[0045] The computer 502 also includes one or more secondary storage devices 514. The secondary storage devices 514 include, for example, a hard disk drive 516 and/or a removable storage device or drive 518. The removable storage drive 518 represents a floppy disk drive, a magnetic tape drive, a compact disk drive, an optical storage device, tape backup, ZIP drive, a slot, a PCMCIA device, etc.

[0046] The removable storage drive 518 interacts with a removable storage unit 520. As will be appreciated, the removable storage unit 520 includes a computer usable or readable storage medium having stored therein computer software (control logic) and/or data. The removable storage drive 518 reads from and/or writes to the removable storage unit 520 in a well known manner.

[0047] Removable storage unit 520, also called a program storage device or a computer program product, represents a floppy disk, magnetic tape, compact disk, optical storage disk, ZIP disk, memory card, PCMCIA card, or any other computer data storage device. Program storage devices or computer program products also include any device in which computer programs can be stored, such as hard drives.

[0048] In an embodiment, the present invention is directed to computer program products or program storage devices having software that enables the computer 502 to perform any combination of the functions described herein.

[0049] Computer programs (also called computer control logic) are stored in main memory 508 and/or the secondary storage devices 514. Such computer programs, when executed, enable the computer 502 to perform the functions of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 506 to perform the functions of the present invention. Accordingly, such computer programs represent controllers of the computer 502.

[0050] The computer 502 also includes a display unit 522, such as a computer monitor, and one or more input devices 524, such as a keyboard, a mouse, other pointing devices (such as a light pen and trackball), etc.

[0051] The computer 502 further includes a communication or network interface 526. The network interface 526 enables the computer 502 to communicate over communication networks, such as networks 108 and 112, which in embodiments use the well known HTTP communication protocol.

[0052] The computer 502 can receive signals 527 from any medium via interface 526. Such signals 527 may include data and/or software. Such signals 527 represent another computer program product embodiment, and the invention is directed to such embodiment.

[0053] The components of the invention (shown in FIG. 1) are described in greater detail below. It should be understood that any specific software, hardware, or operating system implementations described herein are provided for purposes of illustration, and not limitation. The invention can work with software, hardware, and operating system implementations other than those described herein. Any software, hardware, and operating system implementations suitable for performing the functions described herein can be used.

2.2. Databases

[0054] FIG. 3 illustrates an example embodiment of the databases 116. According to an embodiment of the present invention, the databases 116 store documents, information related to the documents, and/or information pertinent to the analysis of the documents.

[0055] FIG. 3 illustrates a particular embodiment of the databases 116, and also illustrates a particular embodiment of the types of tables that the databases 116 contain, and the attributes in the tables. It should be understood, however, that the invention is not limited to the particular database embodiment of FIG. 3. Instead, the invention is adapted and intended to cover other database structures and organizations that are capable of storing documents and information pertinent to the analysis of the documents. The particular documents and information that are stored in the databases are implementation dependent and vary based on a number of factors, including the type of analysis that is desired, the specific needs of the customer, the type and content of the information that the customer maintains, the application, implementation issues, etc.

2.2.1. Document Databases

[0056] The document databases 312 include electronic representations of documents of interest to the customer. The document databases 312 represent the customer's repository of documents, and are thus also called the customer's document repository. (The "repository" could alternatively represent all documents represented in the databases 116, whether represented in the document databases 312 or the bibliographic databases 302.)

[0057] For example, the patent database 314 includes electronic representations of U.S. and foreign patents of interest to the customer. The patent database 314 preferably has stored therein an image file and a text file for each patent represented in the patent database 314, where the image file and the text file are representations of the patent. Details of an embodiment of the image file and the text file are described in U.S. Patent No. 5,623,681 and U.S. Patent No. 5,623,679, which are both incorporated herein by reference in their entireties.

[0058] The document databases 312 also include other documents of interest to the customer.

[0059] The documents in the document databases 312 may be text, images, graphics, audio, video, multimedia, applications, etc.

2.2.2. Document Bibliographic Databases

[0060] The document bibliographic databases 302 store information about documents (as opposed to the documents themselves). More particularly, the document bibliographic databases 302 store bibliographic information about documents.

[0061] For example, the patent bibliographic databases 304 store bibliographic data about U.S. and non-U.S. patents. Such patent bibliographic data includes, but is not limited to, the information on the front page of patents, such as: the patent number, the issue date, the inventors, the title, the assignee, the serial number, the filing date, the U.S. and international classifications, the fields of search, the references cited, the primary examiner, the assistant examiner, the attorney, the agent, the law firm, priority information, related application information, the number of claims, the number of drawing pages, the patent term, the expiration date, etc. The patent bibliographic databases 304 can also include one or more user defined fields that can store large amounts of data.

2.2.3. Notes Database

[0062] The present invention supports annotation of the documents in the document databases 312. More particularly, the present invention allows users to create and link annotations (also called notes) to the documents (or portions thereof) in the document databases 312. Such annotations can include text, graphics, images, video, audio, and/or any other information representation that can be stored in electronic form.

[0063] The annotations, linkage information (i.e., information that specifies the link between a note and a portion of a document), and information related to the annotations and/or the linkage information (such as the position of the linked portion in the document, the date of creation, the creator, the date of modification, a note title and/or subject, access rights, etc.) are stored in the notes databases 340. Embodiments of the notes databases 340 are described in U.S. Patent No. 5,623,679 and U.S. Patent No. 5,623,681, incorporated by reference herein.

2.2.4. Groups Databases

[0064] Information on groups is stored in the group databases 321. Generally, a group is a data structure that includes any number of objects that typically follow a common theme or characteristic (although this is not a mandatory requirement of groups). Groups are said to be document-centric.

[0065] There are two classes of groups: predefined groups (also called system defined groups) and user-defined groups (also called arbitrary groups).

[0066] The invention also supports other types of groups. For example, the invention supports temporary groups. A temporary group is automatically created by the invention in the course of processing a command. One application of temporary groups involves search operations. Specifically, when conducting a search for documents, a new temporary group is created, and the search results are stored in the temporary group. Groups may be created through the processing of other commands or actions. The invention permits operators to save temporary groups. For example, the invention allows operators to convert temporary groups to predefined groups or user-defined groups.

[0067] Groups are further described in U.S. Patent No. 5,991,751, which is herein incorporated by reference in its entirety.

2.3. Example Block Diagram of the Knowledge Management System

[0068] FIG. 2 is a logical block diagram of the KMS 114. The architecture of FIG. 2 is provided for illustrative purposes. Other architectures suitable to achieve the functions of the invention will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein.

[0069] The KMS 114 has a number of modules (collectively called the KMS modules).

[0070] A number of the modules interact with the databases 116. A SQL server 226 (such as the Microsoft SQL Server) and/or other well known database servers 228 interact with the databases 116. The KMS modules interact with these servers 226 and 228 and the databases 116 via a database interface module 220, which may represent, for example, an ODBC (object database connectivity) layer.

[0071] The Network transport layer or interface 201 is used to receive command request objects from the client 104, 106.

[0072] The KMS 114 is a highly secure business decision system. In an embodiment, the specific operations in each command object are checked against the security information maintained about each user in the system. This is logically done through a comprehensive security layer or module 202.

[0073] The document storage and retrieval module 208 provides access to documents and information in the databases 116.

[0074] The Searching subsystem or module 210 provides for document and information searching. The search layer 210 also encapsulates the specific search engine(s) 224 used in the implementation of the system, which can and will vary based on available search technologies.

[0075] The Groups layer or grouping module 212 is responsible for managing groups.

[0076] The Notes layer or module 214 is responsible for managing forms of annotations.

[0077] The Analysis layer or analysis modules 216 perform analysis in support of specific requests made by various modules in the system.

[0078] Finally, the server administration layer or module 218 provides services to manage the configuration of the KMS 114, such as adding or changing the security permissions associated with a specific user.

[0079] The modules described above are further described in U.S. Patent No. 5,991,751, incorporated herein by reference in its entirety.

2.4. Analysis Modules

[0080] Example analysis modules 216 are shown in FIG. 4. These analysis modules 216 semi-automatically and automatically interact and process documents and information contained in the databases 116 pursuant to user commands. Such processing is sometimes referred to herein as "analytics."

The analysis modules 216 have the capability of processing documents in one or more groups, and potentially the parents and/or children of these groups.

[0081] It should be understood that the invention is adapted and intended to include a wide and varied range of analysis modules 216. The analysis modules 216 shown in FIG. 4 represent only a sampling of the analysis modules 216 that the invention is adapted and intended to support. The invention can support many other analysis modules 216 because the databases 116 are so rich. The analysis modules 216 can include any other module that performs useful processing (from the point of view of the customer) of the data in the databases 116.

[0082] Analysis modules 216 are further described in U.S. Patent No. 5,991,751, which is herein incorporated by reference in its entirety.

3. Knowledge Management - Example Work Flow Embodiments

[0083] As its name indicates, the knowledge management system (KMS) 114 is useful for managing knowledge. The knowledge that is being managed comprises documents of interest to users. Such documents may be in any form, such as but not limited to text, images, graphics, audio, video, multimedia, computer programs/applications, etc., and combinations thereof.

[0084] The KMS 114 provides a number of functions for processing and manipulating knowledge. Typically, the functions that are performed are selected by users. More particularly, users select the functions that are to be performed, and the sequence in which those functions are performed. This is referred to as "work flow."

[0085] FIG. 6 illustrates an example work flow 602 using the KMS 114. It is noted that this work flow 602 is provided for illustrative purposes, and is not limiting. There are any number of work flows that can be achieved using the KMS 114. The work flow that is actually used in any given situation depends on a number of factors, such as the knowledge being processed (i.e., the documents being processed), the goals of the task, the application, the tools that are available, etc.

[0086] In step 604 of FIG. 6, one or more searches are performed. These searches can be over databases 116, and/or over another repository of documents, such as over an external database or search engine. The goal of the search is to identify documents which satisfy some criteria (i.e., the search criteria).

[0087] The invention supports a number of searching types. These may be based, for example, on morphological processing 606 (involving the understanding of components of words), lexical processing 608 (matching whole words during searches), syntactical processing 610 (taking into consideration the role of words), semantic processing 612 (the meaning of words in sentences), discourse processing 614 (understanding parts of documents), and/or pragmatic processing 616 (using external knowledge). Based on such processing, the invention supports full text search modules 704, boolean search modules 706, clustering modules 708, as well as other search modules (see FIG. 7).

[0088] The searches of step 604 result in one or more lists of documents. In an embodiment, the lists correspond to groups 618. Accordingly, the searches of step 604 automatically generate groups 618. This is generally true of the invention. That is, processing of documents results in groups, where the groups include documents that were identified, or that resulted, from such processing.

[0089] In an embodiment, the groups 618 are temporary. In such embodiments, it is possible to save the groups 618 (step 620). It is also possible to annotate the groups 618, and the documents in the groups 618 (step 622).

[0090] The user may wish to conduct further searching within the groups 618. Such further searching is represented by step 624, and this results in new groups 626. Again, such groups 626 may be saved (step 628) and/or annotated (step 630).

[0091] In step 632, the user elects to analyze the groups 626 and the documents contained therein. The invention supports a number of analysis modules, such as mapping modules 634, citation or plot lineage modules 636, and reporting modules 638. See also FIG. 4. These modules are further discussed below, as well as in U.S. Patent No. 5,991,751, which is herein incorporated by reference in its entirety. Also consider FIG. 8, which illustrates an embodiment of the citation modules 636.

[0092] The citation modules 636 includes a patent citation module 802 (see U.S. Patent No. 5,991,751), an author citation module 804, and a case law citation module 806.

[0093] The author citation module 804 operates to identify documents related to an author. The operation of the author citation module 804 according to an embodiment is shown in FIG. 9. In step 904, an author is identified or selected. In step 906, searches are conducted over databases 116 (and/or over other repositories of documents) to identify articles written by the author. In step 908, searches are conducted over databases 116 (and/or over other repositories of documents) to identify articles that cite the author. In step 910, the search results of steps 906 and 908 are displayed. A hyperbolic tree mechanism may be utilized for the display (similar to a patent citation map).

[0094] The author citation module 804 as described above is focused on an author. It can be modified to be focused on a particular article (called the based article). For example, references on which the base article is based, and references that cite the base article, could be displayed in step 910.

[0095] The case law citation module 806 operates to display cases on which a given case is based, and cases that cite the given case. In this manner, it is similar to the author citation module 804, and its processing is similar to that shown in FIG. 9.

[0096] Referring again to FIG. 6, the analysis of step 632 results in lists of documents. Again, in an embodiment, those lists correspond to groups 640.

[0097] The processing performed thus far may have been intended to narrow a large database of documents to a smaller set that is pertinent to certain criteria of interest to the user. This involved various searching (steps 604 and 624) and analysis (step 632). The resulting groups 640 may include documents that the user feels are very pertinent to his criteria of interest. Thus, the user is ready at this point to analyze in detail each document in the groups 640. This is represented by step 642. Such analysis may involve a manual study of the documents, and/or further analytics (such as that performed in step 632).

[0098] FIG. 30 is another view of the knowledge management capabilities of the invention. Specifically, FIG. 30 is a state diagram representing some example work flows that are achievable via the KMS 114.

[0099] FIG. 30 indicates that searches 3002 can be performed. Such searches 3002 result in groups 3004. The groups 3004 can be further searched 3006 to thereby generate sub-groups 3008. The groups 3004 can also be analyzed 3010, and such analysis results in groups 3018. Such analysis 3010 may include mapping operations 3012, citation operations 3014, and/or reporting operations 3016.

[0100] Groups and documents can be annotated 3024. Also, documents can be individually analyzed 3020.

[0101] FIG. 37 illustrates another view of the knowledge management capabilities of the invention which is somewhat similar to FIG. 30.

[0102] FIG. 36 depicts another work flow 3626 view of embodiments of the invention. This workflow 3626 indicates an example use of the invention. Specifically, the invention can be used to "funnel" or process a large amount of information so as to achieve "wisdom" 3622, where such wisdom 3622 can have great commercial value 3624. This example indicates that one may utilize commodity online searching 3602 (such as, but not limited to, a search engine available through the Internet) to identify a number of documents somehow related to a topic of interest. Then, through a series of searching and grouping 3604, mapping 3606, visualization relationships 3608, clustering 3610, citations 3612, reports 3614, use of objects 3616, one may identify from the large set of documents a select few that are more particularly related to the topic of interest. This is represented by the "knowledge" block 3620 in FIG.

36. Through analysis of this select set of documents, one can determine a great deal about the topic of interest, resulting in wisdom, which as noted above may have financial value 3624. As noted above, the invention enables users to annotate 3628 the workflow 3626.

[0103] The work flows described herein can be saved. In other words, the invention supports persistent work flows or processes. The invention achieves persistent processes through the use of objects. These objects can be manipulated, as indicated by step 3022 in FIG. 30. This topic is described in the following section.

4. Achieving Persistent Processes Through Use of Objects

[0104] As mentioned above, the invention enables users to save work flows, which are also referred to as processes. In other words, the invention allows users to make processes persistent.

[0105] The invention uses objects to achieve this function. The invention creates objects for components of a process. For example, with regard to the example process of FIG. 6, the invention creates objects for the search operations 604, 624, the groups 618, 626, 640, and the analysis 632 that is performed. The process can thereby be saved by storing such objects.

[0106] A number of advantages are achieved by making a process persistent. Some of the advantages (and features of the invention) are shown in FIG. 11.

[0107] On advantage is a process can be easily fully or partially re-executed by traversing one or more of its objects (i.e., by invoking the objects). This is represented by step 1106.

[0108] Another advantage is it is easy to create new processes by modifying old processes. This is represented by steps 1108-1114. In step 1108, a user copies objects corresponding to an old process to create new objects. In step 110, the user modifies the copied objects to create a new process. In step 1112, the user executes the new process by traversing its objects. The user can then compare the new process to the old process by comparing its respective objects (see step 1114).

[0109] The invention enables users to organize and manage objects. For example, a user can save objects, delete objects, view objects, modify objects, organize objects in folders, etc. Such operation is represented by step 1116.

[0110] The invention supports definitions (or templates) for various objects. The invention supports operation objects (such as query, boolean, import, patent family, corporate family, list exploder objects) and application objects (web reporting, citation tree, patent family tree, corporate family tree, visualization, clustering objects), as well as other objects. As a process is being executed, objects corresponding to the process's components are created (instantiated) based on the applicable object definitions.

[0111] Such operation is depicted in the flowchart 2902 of FIG. 29. In step 2904, an action is selected. This is the action that is to be performed next in the process, such as a search. This action may produce a result, such as a list of documents. In step 2906, the object definitions corresponding to the action and the result are identified, and objects are created based on such object definitions. In step 2908, the action is executed. This may involve updating the objects created in step 2906. For example, at this point, a list object may be populated with the list of documents resulting from executing the action in step 2908 (or the list object could be instantiated at this point). In step 2910, the objects can be saved (this is an optional step).

[0112] Many objects have an input and an output. The input represents one or more other objects, which are called upstream objects 2810 (see the example work flow diagram 2802 in FIG. 28). Similarly, the output represents one or more other objects, which are called downstream objects 2812.

[0113] Thus, the features of the invention just described enable users to create, store, manage, manipulate, and share their work – not just results, but the operations and processes for arriving at those results. In addition, these features enable automation of processes to greatly reduce or eliminate the manual effort required to update previously generated results.

[0114] Each object type has attributes that help users document and re-use their work. Users can create, manage, and access objects from Object Manager, a tree navigation and management tool similar to Windows File Manager.

[0115] With regard to queries (searches), by defining upstream objects (such as a List or database) that will be queried, the Trigger (such as a change to a List or an update to a database source) that could cause the Query to auto-execute, and the downstream object (the output List), a user could set up a number of Queries to feed Lists of common topics (such as competitor activity) that could be shared with perhaps many users. The other types of objects will have similar properties.

[0116] Using the defined objects as building blocks connected together in strings, complete processes can be set up. Some simple examples would be:

Database -> Query -> List -> Report

[List A , List B] -> Boolean -> List C -> ThemeScape

[0117] One powerful feature of this architecture is that these strings of objects can be defined as objects themselves. Called Process Stream Objects, these objects can be created, stored, and auto- or manually-executed. They can also be copied, modified, and moved, greatly simplifying the task of setting up new processes. A goal in developing these features is to provide the building blocks to enable users to construct, store, utilize, and manage whatever Process Streams they might envision.

[0118] Object definitions are described in the following sections. These definitions are provided for illustrative purposes. The invention supports other objects, and the definitions for these other objects will be apparent to persons skilled in the relevant arts based on the teachings contained herein.

4.1.1. Common Object Features

[0119] In an embodiment, the following features apply to all objects.

4.1.1.1. Generic Attributes

[0120] Object Type

The Object Type is a code that identifies the object as a List Object, Import Object, Patent Family Exclude Object, etc. This provides the system with a simple way to verify that connections are attempted only between compatible Object Types.

[0121] Name

The same naming rules apply for all Object Types. A maximum of 50 characters may be used, including any printable ASCII character except < > / \ ', " Naming is case-sensitive; neither the first nor the last character can be a space. When generating a new Object, the system generates a default name of the format (NewList, NewList1, NewList2, etc) and prompts the user for the desired name. The default name is used unless the user enters an alternative.

[0122] Path

The Path for an Object consists of the Windows path to the Object Database, followed by the traversal of the hierarchy (as defined in the Object Database) to the Folder containing the object.

[0123] Object ID

The Object ID is a unique system generated object identifier which allows the system to maintain a simple notation for describing linkages between objects. For example, a list of downstream objects imbedded in the attributes of a Data Object requires only the list of Object IDs, rather than a list of Object Paths/Names. When the user changes the name and/or location of a folder or Data Object, the Object ID stays the same. A lookup table is used to relate Object ID to Object Path/Name. This table is modified if the user moves the object to a different folder, folders in the path are renamed, or folders are moved around in the hierarchy.

[0124] Object Version Number

The Object Version Number indicates the definition and format used when an Object was created. Use of this number allows future changes to the number and type of object attributes, or to the object's storage format. The system should ensure that the object version is current; if not, the system should prompt the user to supply missing attribute information, or insert defaults and re-store in the updated version.

[0125] Object Description

The Object Description is a short user-entered text that can be displayed in the listing of objects in the Object Manager. Maximum text length is 50 characters.

[0126] Owner

The Owner of an object is the user who created it. When objects are copied, the user creating the copy is the Owner of the copy. Ownership implies full read/write/modify/delete/execute permissions. The Owner field is displayable by any user, but is editable only by the system administrator.

[0127] Permissions

In an embodiment, top-level folders are provided for each individual user and each user group; all users who have access to a top-level folder have full access to all objects in that branch of the hierarchy. In another embodiment, permissions may be set separately for individual users, user groups, top-level folders, and individual objects. Supported permissions are Read, Write, Annotate, and Execute.

[0128] Create Date

The Create Date is set to the date the object is created. When an object is copied, the Create Date of the new version is the same as the Create Date of the original. It may be displayed, but cannot be edited.

[0129] Modification Date

The Modification Date is initially set to the Create Date, then is updated based on the following events:

Name changed
Path changed
Owner changed
Object version updated
Permissions changed
Annotations changed
Contents added, deleted, or modified (applies only to objects that have Contents, including Folders and Lists)
Other Attributes changed

Object copied

[0130] Object Modification History

The Object Modification History is a table showing dates the object was modified, the modification that occurred, and the User Name of the user who made the modification. When an object is copied, the Object Modification History of the new version is the same as the Object Modification History of the original, except that the copy has the additional table entry showing that the object was copied. The Object Modification History may be displayed and printed, but not edited.

Object Modification History table entries:

Object created
Name changed to ___
Path changed to ___
Object Description changed
Owner changed to ___
Object version updated to ___
Permissions changed
Annotations changed
Contents added, deleted, or modified
Object copied
Related Objects added or deleted

[0131] Annotations

Objects have annotation capabilities that are similar to the Group annotation capabilities in IPAM.

[0132] Related Objects

Related Objects include Upstream Objects and Downstream Objects in the same Process Stream as the current object:

[0133] Upstream Objects

Upstream Objects are those Data Objects that feed data to the current object. See 2810 in FIG. 28. Object Types that are allowable as upstream Objects depends on the Object Type of the current object. For example, a Query cannot be an upstream Objects for another Query – only Lists and Databases can be upstream Objects for a Query.

[0134] Downstream Objects

Downstream Objects are those Data Objects that receive data from the current object. See 2812 in FIG. 28. Object Types that are allowable as Downstream Objects depends on the Object Type of the current object. For example, a Query cannot be an Downstream Objects for another Query – only Lists can be Downstream objects for a Query.

[0135] Object Triggers

The Object Trigger parameter specifies what causes the object to be executed. Only Operations Objects and Analytic Objects have triggers; List Objects do not. Types of triggers include:

- Time-based triggers (daily, weekly, monthly)
- Event-based triggers (change to an Object, update to a database)
- Manual trigger

4.1.1.2. Generic Object Views

[0136] Permissions View

The Permissions View shows which individual users and user groups have access permissions to the List. These permissions may include Read, Write, and Annotate. The Permissions View also allows the Owner to add, change, and delete permissions for the Object.

[0137] Connections View (see, for example, FIG. 12)

The Connections View shows how the Current Object is connected in the Process Stream to its Upstream and Downstream Objects (see 1202 and 1206 that represent windows in which current connections for the objects are displayed). This view allows the user to add and delete connections to other Objects (see 1204 and 1208 in FIG. 12). The Path/Name of each Object in the Upstream and Downstream Object Lists serves as a hyperlink to Views of that Object. However, these Object Lists are filtered based on the permissions of the user and the Objects, with a placeholder inserted for objects that the user is not permitted to see.

[0138] Trigger View (see, for example, FIG. 13)

The Trigger View allows the user to set up auto-executing Process Streams by selecting the trigger event or execution schedule for the object. In the example of FIG. 13, a query called "NewQuery.qry" (see 1312) is being triggered based on the occurrence of an event (see 1302). The event is a change to an list object called "newproj.lst," and the trigger begins on 3/27/2001 (see 1304), and continues for 10 occurrences thereafter (see 1306).

[0139] Attribute View (see, for example, FIG. 14)

The Attribute View shows the values of the following object attributes:

- Object Type
- Name
- Path
- Object ID
- Description
- Object Version
- Owner
- Create Date
- Modification Date

[0140] Users can change the Description in the Attribute View.

[0141] Modification History View (see, for example, FIG. 15)

The Modification History View provides a table of change activities affecting the List. Date of change and user ID of the user who made the change is also shown. The entire Modification History may be printed from this view.

[0142] Annotation View (see, for example, FIG. 16)

The Annotation View allows the user to view, edit, and print existing List annotations, as well as add new ones. Any user with Annotate permission for the List may add annotations. Annotations of individual Documents contained in the List are handled in the Document View.

4.1.2. Boolean Operation Objects

[0143] The Boolean Operation Object allows users to perform boolean operations on Lists to generate List output. Possible boolean operations are (in the following, the upstream objects are List A and List B, and the downstream object is List C):

AND

List A AND List B -> List C

List C contains all the documents that appear in both List A and List B.

OR

List A or List B -> List C

List C contains all the documents that appear in either List A or in List B.

NOT

List A NOT List B -> List C

List C contains all the documents that appear in List A, but not List B.

See also, for example, FIG. 17.

4.1.3. Corporate Family Operating Objects

[0144] The Corporate Family Operation allows users to define operations based on corporate entity information. For example, this object can sort through a list of patents to identify those that are assigned to a particular corporate entity.

4.1.4. Export Objects

[0145] The Export Operation Object allows tracking of exported Lists and their formats. This object corresponds to an export operation where a particular object, such as a list object for example, is exported to a destination. The export operation includes modifying the object according to the specifications of the destination (such as the format of the destination). The user would be prompted to generate or choose an Export Operation Object when beginning the operation. Examples of Export Operation data transactions are exporting to Lotus Notes or to loosely coupled analytic software.

[0146] Export specific attributes include the following:

[0147] Destination name/provider

[0148] Export Format Definition

Allowed formats could include List, delimited text, spreadsheet, etc. The format definition also specifies which fields are exported, and in what order.

[0149] Export List description

The export list description is a text string that can be provided to an external program that receives the exported list. This description field also allows the user to record the purposes of the export.

4.1.5. Folder Objects

[0150] Folders represent storage locations for other objects, including other Folders. They can be nested in a hierarchy, similar to folders or directories in Windows and DOS. Although in IPAM Folders are implemented in a database rather than in the Windows file system, this is transparent to the user. Folders may be annotated.

[0151] Content View is a Folder-Specific Object View. The Content View shows the contents of the Folder, which can include subfolders and Data

Objects of any type. Contents are listed in alphabetical order by name, with subfolders listed first, followed by Data Objects. In addition to the names of the contents, the listing can optionally include the Object Description. The Content View is the default view. See, for example, FIG. 18.

4.1.6. Import Objects

[0152] The Import Operation Object allows tracking of the source of list imports, as well as reformatting input data into Lists. In cases where the Import Operation is driven by the external source, the external software could generate an Import Operation Object. An example of a data transaction generating an Import Operation externally would be an export from SciFinder to IPAM.

[0153] In cases where the Import Operation is driven from within IPAM, the user would generate an Import Operation Object. An example of an IPAM-driven data transaction could be an import to IPAM.

[0154] Import specific attributes include the following:

[0155] Source name/provider

[0156] Import Format Definition

Accepted formats could include List, delimited text, spreadsheet, etc.

[0157] Import List description

The import list description is a text string that can be provided by an external program that generates the list for import. This might describe the search criteria used to generate the set on another search engine, the name of the list on the source platform, etc.

4.1.7. List Exploder Operation Objects

[0158] The List Exploder Operation allows users to easily break an input List into a number of output Lists based on one of several criteria: Publication Year, Inventor, IPC, subject matter, etc.

4.1.8. List Objects

[0159] A List Object represents a list of documents, implemented as a list of document GUIDs. Display of the List would result in any additional document fields specified for the List View to be fetched from the Patent or Document database. This object corresponds to the concept of "Group" (described elsewhere herein).

[0160]
- Related Objects
  - Permitted Upstream Objects (one object)
- Operation Objects
  - Query
  - Boolean
  - Import
  - Patent Family Dedupe
  - Patent Family Expand
  - Corporate Family
  - List Exploder
- Application Objects
  - Web Reporting
  - Citation Tree
  - Patent Family Tree
  - Corporate Family Tree
  - Themescape
  - ClearForest

[0161] Permitted Downstream Objects (multiple objects)

Operation Objects

- Query
- Boolean
- Export
- Patent Family Dedupe
- Patent Family Expand
- Corporate Family
- List Exploder

Application Objects

- Web Reporting
- Citation Tree
- Patent Family Tree
- Corporate Family Tree
- Claims Tree
- Themescape
- ClearForest

[0162] List-Specific Attributes

Document Table

For each document in the Document List, the following fields are contained in the Document Table in the Object Database:

- Doc ID / GUID
- Document Status Code
- Values: Active, Deleted, Pending-Add, Pending-Delete
- Change Source

The Change Source may be either the Upstream Object or Manual.

- Status Change Date (date of most recent status change)

[0163] Content Change Type

For those changes to the List content that result from the Upstream Object (in contrast to manual Adds and Deletes), the Content Change Type indicates whether additions or deletions of documents must be confirmed by the user. Content Change Type can have a value of either Auto or Pending. Setting the Content Change Type to Pending for an upstream Query, for example, will cause all Adds and Deletes resulting from a refresh run of that Query being initially marked as Pending-Add or Pending-Delete. Upon confirmation by the user, documents marked Pending-Add or Pending-Delete will be changed to Active or Deleted. Setting the Content Change Type to Auto will cause all Adds and Deletes that result from a refresh run of that Query being marked as Active or Deleted.

[0164] List Display Parameters

The List Display Parameters indicate which database fields associated with the documents are to be displayed, the document sort order, and the View and format in which to display the List. These parameters include:

- Default View

  This parameter defines which view is displayed when the List is first opened. Choices are Short List View, Full List View, Abstract List View, and Shoebox List View.
- Max Docs Per Page

  The user may set the number of documents to display on each page of the List. The maximum number varies depending on the List Content View: Short List View (50 docs max per page), Full List View (20), Abstract List View (10), and Shoebox List View (5).

- Sort Order
  The Sort Order parameter indicates the primary, secondary, and tertiary sort keys for displaying the List. Any of the biblio fields may be selected as a sort key.

- Display Fields
  A series of check boxes in the List Option View allows the user to select which fields will be displayed for each of the List Content Views. Fields include biblio fields, abstract, front page thumbnail image, or primary drawing. The specific fields available vary between the different List Content Views.

- Status Code Display
  Allows the user to choose Document Status Codes for which documents will be displayed. For example, the user may choose to hide all Deleted documents; or may display only the Pending-Add and Pending-Delete documents.

- Document View Links
  Allows the user to choose which Document View Links to display. Document Views include Standard View, Fulltext, Claims, Summary Page, HTML Image, PDF Image,

Derwent WPI record, etc. See Document View Options MRD for detailed information.

List-Specific Views

[0165] List Display Options View

In the List Display Options View, the user may define the Default View for displaying the List, the fields to display, the fields to sort on, and the number of documents to display per page. The user may also choose to display or hide documents for each of the Document Status Codes. The default object view is either 1) the default view defined for that particular List, or 2) the default view specified in the user's Preference setting. Priority of 1) or 2) is specified in the user's Preference setting. See, for example, FIG. 19.

[0166] List Content Views:

The List Content Views show the contents of the List, including selected database fields related to the document. Depending on the specific List Content View chosen, the following fields may be available for display:

- Country Code
- Doc ID

  The Doc ID serves as a hyperlink to the Default View for that document.
- Kind Code
- Priority Date
- Filing Date
- Pub Date
- Assignee
- Inventor

- IPC Code
- USPC Code
- Legal Status
- Title
- Abstract
- Front Page Thumbnail image
- Primary Drawing
- Document Status Codes
  Document data is displayed in the font color associated with its Status Code; i.e., black for Active, gray for Deleted, green for Pending-Add, red for Pending-Delete.
- Change Source
  The Change Source, when it is the Upstream Object, serves as a hyperlink to the Default View for that object.
- Status Change Date
- Document View Links
  Icons appear only for those Document Views that are available for that particular document.

List Content Views include:

[0167] Short List View (see, for example, FIG. 20)
The Short List View shows a table with one line per document. Rolling the mouse over cells in the table will cause a pop-up to appear showing the entire contents of the cell. This view maximizes the number of records that can appear on each page. Abstracts, drawings, and thumbnails may not be displayed in

this view. A maximum of 50 documents per page are allowed in this view.

[0168] Full List View (see, for example, FIG. 21)

The Full List View shows a table with contents of each cell wrapping to as many lines as necessary to show the entire contents of the cell. Abstracts, drawings, and thumbnails may not be displayed in this view. A maximum of 20 documents per page are allowed in this view.

[0169] Abstract List View

The Abstract List View can show the greatest detail regarding each document, and is used for browsing the abstract and other fields. This is the only view in which the abstract is available. A maximum of 10 documents per page are allowed in this view.

[0170] Shoebox List View (see, for example, FIG. 22)

The Shoebox List View is used for browsing thru Primary Drawings. If no Primary Drawing is available, the Front Page Thumbnail is presented. The fields available for display in this view are very limited to conserve screen space and allow rapid page loading. A maximum of 5 documents per page are allowed in this view.

4.1.9. Query Objects

[0171] Query operations may be executed against any supported database/search engine. Depending on the database/search engine, queries may be structured in searchscreen format and/or native query strings. For example, an initial Query Object implementation for a Dialog search could provide a screen for entering the Dialog search command string; a later implementation could provide a search form that automatically generates Dialog command strings. See, for example, FIG. 23.

[0172] Query specific attributes include the following:

Database name/provider

[0173] Query string

The query string is preferably in a displayable/modifiable format – i.e., displayed to the user as a filled-in search form showing fields available in the subject database, and/or as a native query string. Native query strings would be SQL for queries to SQLServer databases, Dialog command language for Dialog queries, etc.

4.1.10. Patent Family Dedupe

[0174] The Patent Family Dedupe operation allows users to dedupe a List of patent documents so as to keep only one member of a patent family in the result List. For example, if a list includes a U.S. patent and its PCT, Japanese, and European counterparts, this operation allows the user to delete all copies of this patent except for one.

[0175] Users will select the Family Definition that they wish to use (Inpadoc Family, Derwent Family, Identical Priority Family, or Priority-in-Common Family). Users also select Document Retention Priority, which is the priority order for keeping documents so that the retained doc from each family contains the maximum amount of useful information. An example priority order might be to retain the patents according to the following preference: 1) WPI record, 2) US grant, 3) EP-B publication in English, 4) PCT publication in English, etc.

[0176] This object has the following object specific views:

Document Retention Priority View

[0177] The Document Retention Priority View allows users to select the priority order for keeping documents so that the retained doc from each family contains the appropriate amount of useful information in the preferred language. For example, if the Output List is to be reviewed by managers who prefer only to read abstracts, then the priority order would be chosen to provide the highest quality abstract, without regard to availability of fulltext. However, if the reviewer is a scientist or engineer, the priority order might reflect the requirement for fulltext. An example priority order might be: 1) WPI record, 2) US grant, 3) EP-B publication in English, 4) PCT publication in English, etc. See, for example, FIG. 24.

Family Definition View

[0178] The Family Definition View allows the user to select their preferred Family Definition. Possible definitions include the standard definitions: Inpadoc Family or Derwent Family; or user-defined families based on either Identical Priority or Priority-in-Common. For user-defined families, the user must select the specific document-to-document relationships that the user considers to be a Family Relationship. See, for example, FIG. 25.

4.1.11. Patent Family Expand Object

[0179] The Patent Family Expand operation allows users to expand a List of patent documents so that for each patent family represented in the Input List, all family members are included in the Result List. For example, if a List includes a particular U.S. patent, this operation causes all counterpart patent applications and patents to be inserted into the List.

[0180] Membership in a family is determined by the user's choice of Family Definition (Inpadoc Family, Derwent Family, Identical Priority Family, or

Priority-in-Common Family). The user may also choose which types of documents to include, i.e., Pseudo-Docs, Primary Patent Docs, and/or Secondary Patent Docs.

[0181] This object has an object-specific view called the Document Inclusion View. The Document Inclusion View allows the user to select the Input List, Result List, Document Types to include, and the preferred Family Definition. Possible Family Definitions include the standard definitions (Inpadoc Family or Derwent Family), or custom Family Definitions (based on either Identical Priority or Priority-in-Common). For custom Family Definitions, the user must select the specific document-to-document relationships that the user considers to be a Family Relationship. See, for example, FIG. 26.

4.2. Work Flow (Process) Example

[0182] FIGS. 10A and 10B illustrate an example work flow diagram 1002 that shall be used to further illustrate the object features of the invention. This diagram 1002 indicates the objects that are created for components of the work flow.

[0183] In step 1006, a query is conducted on a database 1004. A query object is created for the query.

[0184] In step 1008, List A results from the query, and a list object corresponding to List A is created.

[0185] In step 1010, duplicate patents are removed from List A. A patent family dedupe object is created corresponding to this operation.

[0186] In step 1012, List C results from the operation of step 1010, and a list object corresponding to List C is created.

[0187] In step 1016, documents are imported from some source 1014. This creates an import object.

[0188] In step 1018, List B results from the operation of step 1016, and a list object corresponding to List B is created.

[0189] In step 1020, a boolean operation is performed on List C and List B. This results in a boolean operation object.

[0190] In step 1022, List D results from the operation of step 1020, and a list object corresponding to List D is created.

[0191] In step 1024, a corporate family analysis is performed to identify patents contained in List D that are assigned to a particular corporate entity. This results in a corporate family operating object.

[0192] In step 1026, List E results from the operation of step 1026, and a list object corresponding to List E is created.

[0193] Referring now to FIG. 10B, in step 1028, List E is divided into multiple lists according to some criteria selected by the user. This results in a list exploder operation object.

[0194] As a result of step 1028, List F (step 1030) and List G (step 1032) are created. This results in list objects corresponding to Lists F and G.

[0195] In step 1034, some report is performed on List G. This results in an application object being created.

[0196] In step 1036, List H results from the operation of step 1034, and a list object corresponding to List H is created.

[0197] In step 1038, a mapping operation is performed on the List H. This results in an application object being created.

[0198] In step 1040, List I results from the operation of step 1038, and a list object corresponding to List I is created.

[0199] In step 1042, List I is exported to some destination 1044. This results in an export object being created.

[0200] The work flow of FIGS. 10A and 10B is then complete. The objects generated during the work flow may be save and then re-used at a later time, as described above.

5. Object Manager and Database

[0201] Objects are stored in an object database. The invention also provides an object manager for manipulating objects.

[0202] The Object Manager is a tool for visualizing the Folder hierarchy within the Object Database. See, for example, FIG. 27. When the Object Manager is open, the page is framed, dividing the screen into three areas:

(1) the header / menu bar area at the top;

(2) the Object Manager pane on the left, displaying the folder hierarchy as an indented outline;

(3) the Work Pane on the right, which is used for a variety of tasks and objects.

[0203] When the Object Manager is active and a folder is selected, the Work Pane displays the folder contents. When the Object Manager is active and an object is selected, the Work Pane displays that object's Default View.

6. Example Modules of the Invention

6.1. Clustering Tools

[0204] Referring to FIG. 7, the invention supports various clustering modules for analyzing documents. Such clustering modules organize documents into meaningful groups. More specifically, such clustering modules analyze documents and organize them into a hierarchy or tree according to their content (using, for example, a well-known "Windows Explorer"-style interface).

[0205] Generally, some embodiments of the clustering modules operate by parsing the documents to extract particular information. The information may be extracted from unstructured data contained in the document (such as the specification of a patent), and/or from structured data contained in the document (such as meta data, i.e., titles, authors, abstracts, etc.). The documents are then grouped according to this information. The invention supports clustering modules that operate differently than just described.

[0206] FIG. 31 illustrates an example process (work flow) diagram 3102 that involves a cluster module. In step 3106, a query is made over a database 3104. This results in a list 3108 of documents. Suppose that the query was directed to CDMA (code division multiple access). In this case, the list 3108 would include documents that related in some way to CDMA.

[0207] However, the list 3108 could include hundreds or thousands (or more) of documents, all relating in some way to CDMA. The user might only be interested in a particular CDMA topic, such as GSM.

[0208] In the example of FIG. 31, in step 3110, the list 3108 is processed by a clustering module. The clustering module operates to analyze the documents in the list and assign them to categories according to their content (as represented by their unstructured and/or structuured portions). In an embodiment, each of these categories represents a group. The categories may have sub-categories, each of which also represents a group (or sub-group). Accordingly, the clustering module produces from the list 3108 hierarchically ordered groups 3112. These groups can then be processed as elsewhere described herein. An example of this hierarchical tree is shown in FIG. 32.

[0209] An example of a clustering module that operates as just described is Vivisimo of Pittsburgh, Pennsylvania, although the invention is not limited to this example.

6.2. Relevancy Visualization Tool

[0210] The invention supports other modules for processing and organizing documents. For example, the invention supports a relevancy visualization (RV) module that extracts key terms from documents (from unstructured and/or structured portions of the documents), assigns them to meaningful categories (a taxonomy), and establishes their inter-relationships. As a result, the RV module generates a highly structured body of information that users can slice as needed. Users can generate patterns related to the documents in a variety of visual forms such as maps, tables and graphs.

[0211] More particularly, the RV module analyzes a group of documents to identify how those documents relate to various key terms, and/or relate to each other with regard to (or relative to) those key terms. For example, suppose the key term is "assignee." In this patent related example, the RV module would analyze a group of patents to determine who their assignees are. The RV performs this task by performing a key word search of the documents, for example.

[0212] FIG. 34 illustrates an example work flow diagram involving the RV module. In step 3404, the RV module analyzes a group 3402 of documents according to key terms that the user previously supplied. In particular, the RV module determines the relevance of these key terms on the documents. The output of the RV module is displayed in step 3408. An example display is shown in FIG. 35.

[0213] In embodiments, the RV module operates according to a rule book 3406. The rule book comprises a set of instructions describing specific linguistic patterns relevant to a particular vertical market or horizontal discipline, such as patents. The rulebook tells the module what concepts and relationships to retrieve from the text documents.

[0214] In an embodiment, the rule book 3406 includes rules to analyze patents. The rule book may include instructions to analyze patents according to the following key terms: assignee and inventor. When processing a group of patents (as shown in the example of FIG. 34), the RV module 3404 would analyze the patents in the group 3402 to identify the assignees and the inventors of the patents. This information would then be displayed in step 3408, as shown for example in FIG. 35.

[0215] FIG. 35 indicates that document D1 has inventor I1 and assignee CE1. Document D2 has inventors I1 and I2 and assignee CE1. Document D3 has inventor I2 and assignee CE2.

[0216] An example of this module is ClearForest of New York, NY, although the invention is not limited to this example.

6.3. Patent Family Expand Features

[0217] This section describes features of the invention relating to processing patent families. A patent family is a collection of patent documents that are related to one another in some way. For example, a patent family may include a U.S. provisional application, a corresponding U.S. non-provisional application, a corresponding PCT application, and a Japanese patent that was filed from the PCT application.

Requirements

[0218] This section describes requirements needed to support the patent family features according to an embodiment of the invention.

[0219] Family Relationship Types

The following family relationship types must be represented in the database. Other relationship types may be added based on analysis of each country's patenting process.

[0220] Unstructured Family Relationships

Unstructured Family Relationships do not provide specifics of how one document is related to another – only that a relationship exists. Priority data is contained in many databases. For example, tables of family members are contained in Inpadoc PFS and Derwent WPI. Technical Equivalents are patent documents that cover the same invention as other members of the family, but for a variety of reasons are not linked thru common priorities. Following are the Unstructured Family Relationship types:

- Priority
- Technical Equivalent Family Member
- Inpadoc Family
- Derwent Family

[0221] Structured Family Relationships

Structured family relationships describe the specifics of how one document is related to another in a family. These relationships are programmatically derived based on knowledge of each country's patenting process, numbering system, and database fields. For example, an EP grant has the same Doc ID as its corresponding application, except that the kind code is different. Following is a list of some Structured Family Relationship types.

- Application to Priority
- Grant to Priority
- Continuation in Part to Application
- Post-Issuance to Application
- Eg: Supplemental Search Reports, Corrections, etc
- Grant to Application

- Post-Issuance to Grant
- Re-exam of Grant
- Reissue of Grant
- Certificate of Correction to Grant
- EP National Transfers (R1, R4) based on EP Regional Grant
- National Level Translations (T) based on EP Grant
- Transfer of Application
  Eg: PCT Transfers (W1, W4), EPO National Application Transfers

[0222] Family Definitions

For a variety of functions supported by the invention, the user chooses the Family Definition desired. Family Definitions include Standard and Custom Families.

[0223] Standard Family Definitions

- Inpadoc Family
  The Inpadoc Family links documents appearing in the family table in the Inpadoc PFS database. These families are based on Priority-In-Common, but also have some Technical Equivalent family members as identified by Inpadoc/EPO.

- Derwent Family
  The Derwent Family links documents appearing in the family table in the Derwent WPI database.

[0224] Custom Family Definitions

Family Definitions are fundamentally either Identical Priority or Priority-In-Common. However, in an embodiment these definitions also allow the user to select which Family Relationship Types to include or exclude. For example, a user may choose to exclude all post-issuance documents so as to focus only on Primary Patent Documents.

- Identical Priority Family
  This definition requires that all family members have exactly the same set of priority documents.
- Priority-in-Common Family
  This definition includes in the family any document that has at least one priority document in common with other docs in the family.

[0225] Document Types

In order to facilitate the Usages below, documents within the system are identified according to a Document Type. The following Document Types will be represented:

- Pseudo-Docs
  This is a classification for non-issued documents, such as US applications prior to 2001. A Pseudo-Doc has data associated with it (eg US application number, application date, and family relationships), but is not actually an issued document.
- Primary Patent Docs
  This includes publications of Applications, Grants, Reissues. Primary Patent Docs would be identified as such in Lists.
- Secondary Patent Doc

This includes post-issuance docs such as Supplementary Search Reports (EP-A3, EP-A4), Certificates of Correction, etc.

Usage of Patent Family Data

[0226] Family Report

A Family Report is a listing of the family members for a particular document. In embodiments, this report requires the patent family table data from Inpadoc, and can present only the Inpadoc Patent Family. In other embodiments, as Derwent Family data is acquired and Family RelationshipTypes are identified, the Family Report will be able to present all Family Definitions.

[0227] Family Tree

A Family Tree is a graphical representation of the relationships between each of the documents in the family. Users choose the Family Definition desired before the graph is drawn. A Family Tree example is shown in FIG. 33.

[0228] Patent Family Expand

The Patent Family Expand operation allows users to expand a List of patent documents so that for each patent family represented in the Input List, all family members are included in the Result List. Membership in a family is determined by the user's choice of Family Definition (Inpadoc Family, Derwent Family, Identical Priority Family, or Priority-in-Common Family). The user may also choose which types of documents to include, ie Pseudo-Docs, Primary Patent Docs, and/or Secondary Patent Docs. This is further described above.

[0229] Patent Family Dedupe

The Patent Family Dedupe operation allows users to dedupe a List of patent documents so as to keep only one member of a patent family in the Result List. Users select the Family Definition that they wish to use (Inpadoc Family, Derwent Family, Identical Priority Family, or Priority-in-Common Family). Users also select Document Retention Priority, which is the priority order for keeping documents so that the retained doc from each family contains the maximum amount of useful information. An example priority order might be: 1) WPI record, 2) US grant, 3) EP-B publication in English, 4) PCT publication in English, etc. Only Primary Patent Documents are retained. This is further described above.

Conclusion

[0230] While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example only, and not limitation. Thus, the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.